Airport rhapsody

The described activities take place at an airport, somewhere in Portugal, and aim to portray what happens when passengers arrive from a flight. There are eight main locations: the arrival lounge, the baggage collection point, the temporary storage area (for holding the luggage of passengers in transit), the baggage reclaim office, the terminal transfer quays, the arrival terminal exit and the departure terminal entrance.

There are three types of entities: the *passengers*, who terminate their voyage at the airport or are in transit, the *porter*, who unloads the the bags from a plane, when it lands, and carries them to the baggage collection point or to the temporary storage area, and the *bus driver*, who moves the passengers in transit between the arrival and the departure terminals.

K plane landings are assumed, each involving the arrival of *N* passengers. Each passenger carries 0 to *M* pieces of luggage in the plane hold. The bus, which moves the passengers between terminals, has a capacity of *T* seating places.

Activities are organized, for each plane landing, in the following way

- the passengers walk from the arrival lounge to the baggage collection point, if their journey ends at this airport and have bags to collect; those without bags go directly to the arrival terminal exit and leave the airport; the remaining passengers, who are in transit, walk to the terminal transfer quay;
- after all the passengers have left the plane, the porter picks up the pieces of luggage, one by one, from the plane hold and carries them either to the baggage collection point, or to the temporary storage area, as they belong to local or in transit passengers, respectively;
- in the baggage collection point, the passengers wait for the arrival of their bags; upon taking possession of them, they walk to the arrival terminal exit and leave the airport; those with missed bags go first to the baggage reclaim office to post their complaint, before walking to the arrival terminal exit and leave the airport;
- on the terminal transfer quay, the passengers wait for the bus arrival, which will take them to the departure terminal for the next leg of the journey;
- the bus leaves the terminal transfer quay according to a predefined schedule, executing a circular path which has as another stop the terminal transfer quay of the departure terminal; however, if it happens that all seats are occupied prior to the predefined time to leave, the driver may depart sooner.

In the end of the day, a full report of the activities is issued.

Assume that there are five plane landings, each involving the arrival of six passengers, carrying a maximum of two pieces of luggage in the plane hold and that the transfer bus has a capacity of three seating places. Write a simulation of the life cycle of the passengers, the porter and the bus driver using the client-server model with server replication, where the passengers, the porter and the bus driver are the *clients* and access to the information sharing regions are the services provided to them by the *servers*.

The operations that were previously assigned to activities carried out in the information sharing regions (for the already implemented concurrent version), must now be assigned to independent requests performed on the servers through message passing. In each case, a connection has to be established, a request has to be made, waiting for the reply will follow and the connection has to be closed.

One aims for a solution to be written in Java, to be run in Linux under TCP sockets in a concentrated manner (on a single platform) and to terminate (it must contemplate service shutdown). A *logging* file, that describes the evolution of the internal state of the problem in a clear and precise way, must be included.

Guidelines for solution implementation

1. Specify for each representative server of an information sharing region the structure of the messages to be exchanged.

2. Specify the general organization of the servers architecture.

3. Specify the general organization of the clients architecture.

4. Sketch the interaction diagram which describes in a compact, but precise, way the dynamics of your solution. Go back to steps 1, 2 and 3 until you are satisfied the description is correct.

5. Proceed to its coding in Java as specific reference data types.

6. Specify the mapping of the servers and the clients on a single platform and write the shell scripts which enable the deployment and the execution of the different modules the application is composed of.

7. Validate your solution by taking several runs and checking for each, through the detailed inspection of the logging file, that the output data is indeed correct.